Strings of Consecutive Happy Numbers
1 March 2008
15 Aug 2008
24 Sept 2008
25 Mar 2009
11 July 2009

We need a procedure, given n, to find the minimal N such that S(N) = n.
In our text explanations, we will denote this minimal N by L(n).

```
> restart;
> f := n → n^2;  #in case we ever want to investigate the cube of the digits, etc.
```

$$f := n \rightarrow n^2 \quad \textbf{(1)}$$

```
> bs := 10;
      #this is the base, in case we ever want to investigate binary or ternary or any other base.
```

$$bs := 10 \quad \textbf{(2)}$$

```
> onestep := proc(n1)
      #this is what Dr. Grundman calls S_2(n1) and what we will simply call S below.
  local ans, n, d;
  n := n1;
  ans := 0;
  while n > 0 do
  d := n mod bs;
  ans := ans + f(d);
  n := (n-d)/bs;
  end do;
  ans;
  end;
```

```
onestep := proc(n1)                                                    (3)
    local ans, n, d;
    n := n1;
    ans := 0;
    while 0 < n do d := mod(n, bs); ans := ans + f(d); n := (n - d)/bs end do;
    ans
end proc
```

```
> happy := proc(n)
  local m, j, height;
  m := n;
  height := -1;
  for j from 1 to 100 while (m > 1 and m ≠ 4) do
  m := onestep(m);
   end do;
  if m = 1 then height := j; end if;
  height;
```

```
    end;

happy := proc(n)                                                    (4)
    local m, j, height;
    m := n;
    height := -1;
    for j to 100 while 1 < m and m <> 4 do m := onestep(m) end do;
    if m = 1 then height := j end if;
    height
end proc
```

Given the values for L(k) for all k<n, we will bootstrap ourselves to find the next value of L(n). To do this, we try all possible final digits of L(n), and take the best candidate. First, we need a procedure that will add the trial digit to the number and sort the digits.

```
> AddSortDigits := proc(a, d)
  local b, e, i;
  b := a;
  e := [ ];
  while b > 0 do e := [ b mod bs, op(e) ]; b := iquo(b - (b mod bs), bs); end do;
  e := [op(e), d];
  e := sort(e);
  b := 0;
  for i from 1 to nops(e) do
  b := b·bs + e[i];
  end do;
  b;
  end;
AddSortDigits := proc(a, d)                                         (5)
    local b, e, i;
    b := a;
    e := [ ];
    while 0 < b do e := [mod(b, bs), op(e)]; b := iquo(b - (mod(b, bs)), bs) end do;
    e := [op(e), d];
    e := sort(e);
    b := 0;
    for i to nops(e) do b := b * bs + e[i] end do;
    b
end proc
```

```
> AddSortDigits(36436, 4)
                                    334466                          (6)
```

Now we initialize our list of least values.

```
> A := [1, 11, 111, 2, 12, 112, 1112, 22, 3];
                    A := [1, 11, 111, 2, 12, 112, 1112, 22, 3]      (7)
> for n from 10 to 15 do
  L := AddSortDigits(A[n - 1], 1);
  for d from 2 to 3 do
  M := AddSortDigits(A[n - onestep(d)], d);
```

```
if M < L then L := M; end if;
end do;
A := [op(A), L];
end do:
A;
```

[1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123] **(8)**

```
> A := [op(A), 4];
```

A := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4] **(9)**

```
> for n from 17 to 24 do
  L := AddSortDigits(A[n - 1], 1);
  for d from 2 to 4 do
  M := AddSortDigits(A[n - onestep(d)], d);
  if M < L then L := M; end if;
  end do;
  A := [op(A), L];
  end do:
  A;
```

[1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224] **(10)**

```
> A := [op(A), 5];
```

A := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5] **(11)**

```
> for n from 26 to 35 do
  L := AddSortDigits(A[n - 1], 1);
  for d from 2 to 5 do
  M := AddSortDigits(A[n - onestep(d)], d);
  if M < L then L := M; end if;
  end do;
  A := [op(A), L];
  end do:
  A;
```

[1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135] **(12)**

```
> A := [op(A), 6];
```

A := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6] **(13)**

```
> for n from 37 to 48 do
  L := AddSortDigits(A[n - 1], 1);
  for d from 2 to 6 do
  M := AddSortDigits(A[n - onestep(d)], d);
  if M < L then L := M; end if;
  end do;
  A := [op(A), L];
  end do:
> A := [op(A), 7];
```

A := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6, 16, 116, 1116, 26, 45, 145, 335, 226, 36, 136, 1136, 444, 7] **(14)**

```
> for n from 50 to 63 do
    L := AddSortDigits(A[n - 1], 1);
    for d from 2 to 7 do
    M := AddSortDigits(A[n - onestep(d)], d);
    if M < L then L := M; end if;
    end do;
    A := [op(A), L];
    end do:
> A := [op(A), 8];
```

$A := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6, 16, 116, 1116, 26, 45, 145, 335, 226, 36, 136, 1136, 444, 7, 17, 117, 46, 27, 127, 1127, 246, 227, 37, 137, 1137, 56, 156, 1156, 8]$ **(15)**

```
> for n from 65 to 80 do
    L := AddSortDigits(A[n - 1], 1);
    for d from 2 to 8 do
    M := AddSortDigits(A[n - onestep(d)], d);
    if M < L then L := M; end if;
    end do;
    A := [op(A), L];
    end do:
> A := [op(A), 9];
```

$A := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6, 16, 116, 1116, 26, 45, 145, 335, 226, 36, 136, 1136, 444, 7, 17, 117, 46, 27, 127, 1127, 246, 227, 37, 137, 1137, 56, 156, 1156, 8, 18, 118, 337, 28, 128, 356, 1356, 66, 38, 57, 157, 266, 238, 257, 1257, 48, 9]$ **(16)**

To find how far we need to go we use our maximal length single-digit numbers 111222334445666777888888

```
> onestep(111222334445666777888888)
```

$$809 \quad \textbf{(17)}$$

```
> for n from 82 to 900 do
    L := AddSortDigits(A[n - 1], 1);
    for d from 2 to 9 do
    M := AddSortDigits(A[n - onestep(d)], d);
    if M < L then L := M; end if;
    end do;
    A := [op(A), L];
    end do:
> A;
```

$[1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6, 16, 116, 1116, 26, 45, 145, 335, 226, 36, 136, 1136, 444, 7, 17, 117, 46, 27, 127, 1127, 246, 227, 37, 137, 1137, 56, 156, 1156, 8, 18, 118, 337, 28, 128, 356, 1356, 66, 38, 57, 157, 266, 238, 257, 1257, 48, 9, 19, 119, 248, 29, 129, 1129, 466, 58, 39, 139, 1139, 258, 239, 1239, 448, 49, 77, 177, 68, 168, 277, 1277, 268, 458, 59, 159, 666, 368, 259, 1259, 2666, 78, 178, 359, 468, 69, 169, 1169, 2468, 269, 378, 577, 1577, 568, 369, 1369, 88, 188, 79, 179, 288, 469, 279, 1279,$ **(18)**

668, 388, 578, 379, 1379, 2388, 569, 1569, 488, 89, 189, 777, 1777, 289, 1289, 2777, 4668, 588, 389, 579, 1579, 2588, 2389, 2579, 4488, 489, 99, 199, 688, 1688, 299, 1299, 2688, 4588, 589, 399, 1399, 3688, 2589, 2399, 12399, 788, 499, 779, 1779, 689, 1689, 2779, 12779, 2689, 3788, 599, 1599, 5688, 3689, 2599, 888, 1888, 789, 1789, 2888, 4689, 699, 1699, 6688, 3888, 2699, 3789, 5779, 15779, 5689, 3699, 4888, 889, 1889, 799, 1799, 2889, 4699, 2799, 12799, 5888, 3889, 5789, 3799, 13799, 23889, 5699, 15699, 4889, 899, 1899, 6888, 16888, 2899, 12899, 26888, 45888, 5889, 3899, 5799, 15799, 25889, 23899, 25799, 7888, 4899, 999, 1999, 6889, 16889, 2999, 12999, 26889, 37888, 5899, 3999, 13999, 36889, 25899, 8888, 18888, 7889, 4999, 7799, 17799, 6899, 16899, 27799, 38888, 26899, 37889, 5999, 15999, 56889, 36899, 25999, 8889, 18889, 7899, 17899, 28889, 46899, 6999, 16999, 58888, 38889, 26999, 37899, 57799, 157799, 56899, 36999, 48889, 8899, 18899, 7999, 17999, 28899, 46999, 27999, 127999, 58889, 38899, 57899, 37999, 137999, 238899, 56999, 78888, 48899, 8999, 18999, 68889, 168889, 28999, 128999, 268889, 378888, 58899, 38999, 57999, 157999, 258899, 88888, 188888, 78889, 48999, 9999, 19999, 68899, 168899, 29999, 129999, 268899, 378889, 58999, 39999, 139999, 368899, 258999, 88889, 188889, 78899, 49999, 77999, 177999, 68999, 168999, 277999, 388889, 268999, 378899, 59999, 159999, 568899, 368999, 259999, 88899, 188899, 78999, 178999, 288899, 468999, 69999, 169999, 588889, 388899, 269999, 378999, 577999, 1577999, 568999, 369999, 488899, 88999, 188999, 79999, 179999, 288999, 469999, 279999, 1279999, 588899, 388999, 578999, 379999, 1379999, 888888, 569999, 788889, 488999, 89999, 189999, 688899, 1688899, 289999, 1289999, 2688899, 3788889, 588999, 389999, 579999, 1579999, 2588999, 888889, 1888889, 788899, 489999, 99999, 199999, 688999, 1688999, 299999, 1299999, 2688999, 3788899, 589999, 399999, 1399999, 3688999, 2589999, 888899, 1888899, 788999, 499999, 779999, 1779999, 689999, 1689999, 2779999, 3888899, 2689999, 3788999, 599999, 1599999, 5688999, 3689999, 2599999, 888999, 1888999, 789999, 1789999, 2888999, 4689999, 699999, 1699999, 5888899, 3888999, 2699999, 3789999, 5779999, 8888888, 5689999, 3699999, 4888999, 889999, 1889999, 799999, 1799999, 2889999, 4699999, 2799999, 12799999, 5888999, 3889999, 5789999, 3799999, 13799999, 8888889, 5699999, 7888899, 4889999, 899999, 1899999, 6888999, 16888999, 2899999, 12899999, 26888999, 37888899, 5889999, 3899999, 5799999, 15799999, 25889999, 8888899, 18888899, 7888999, 4899999, 999999, 1999999, 6889999, 16889999, 2999999, 12999999, 26889999, 37888999, 5899999, 3999999, 13999999, 36889999, 25899999, 8888999, 18888999, 7889999, 4999999, 7799999, 17799999, 6899999, 16899999, 27799999, 38888999, 26899999, 37889999, 5999999, 15999999, 56889999, 36899999, 25999999, 8889999, 18889999, 7899999, 17899999, 28889999, 46899999, 6999999, 16999999, 58888999, 38889999, 26999999, 37899999, 57799999, 88888889, 56899999, 36999999, 48889999, 8899999, 18899999, 7999999, 17999999, 28899999, 46999999, 27999999, 127999999, 58889999, 38899999, 57899999, 37999999, 137999999, 88888899, 56999999, 78888999, 48899999, 8999999, 18999999, 68889999, 168889999, 28999999, 128999999, 268889999, 378888999, 58899999, 38999999, 57999999, 157999999, 258899999, 88888999, 188888999, 78889999, 48999999, 9999999, 19999999, 68899999, 168899999, 29999999, 129999999, 268899999, 378889999,

58999999, 39999999, 139999999, 368899999, 258999999, 88889999, 188889999, 78899999, 49999999, 77999999, 177999999, 68999999, 168999999, 277999999, 388889999, 268999999, 378899999, 59999999, 159999999, 568899999, 368999999, 259999999, 88899999, 188899999, 78999999, 178999999, 288899999, 468999999, 69999999, 169999999, 588889999, 388899999, 269999999, 378999999, 577999999, 888888899, 568999999, 369999999, 488899999, 88999999, 188999999, 79999999, 179999999, 288999999, 469999999, 279999999, 1279999999, 588899999, 388999999, 578999999, 379999999, 1379999999, 888888999, 569999999, 788889999, 488999999, 89999999, 189999999, 688899999, 1688899999, 289999999, 1289999999, 2688899999, 3788889999, 588999999, 389999999, 579999999, 1579999999, 2588999999, 888889999, 1888889999, 788899999, 489999999, 99999999, 199999999, 688999999, 1688999999, 299999999, 1299999999, 2688999999, 3788899999, 589999999, 399999999, 1399999999, 3688999999, 2589999999, 888899999, 1888899999, 788999999, 499999999, 779999999, 1779999999, 689999999, 1689999999, 2779999999, 3888899999, 2689999999, 3788999999, 599999999, 1599999999, 5688999999, 3689999999, 2599999999, 888999999, 1888999999, 789999999, 1789999999, 2888999999, 4689999999, 699999999, 1699999999, 5888899999, 3888999999, 2699999999, 3789999999, 5779999999, 8888888999, 5689999999, 3699999999, 4888999999, 889999999, 1889999999, 799999999, 1799999999, 2889999999, 4699999999, 2799999999, 12799999999, 5888999999, 3889999999, 5789999999, 3799999999, 13799999999, 8888889999, 5699999999, 7888899999, 4889999999, 899999999, 1899999999, 6888999999, 16888999999, 2899999999, 12899999999, 26888999999, 37888899999, 5889999999, 3899999999, 5799999999, 15799999999, 25889999999, 8888899999, 18888899999, 7888999999, 4899999999, 999999999, 1999999999, 6889999999, 16889999999, 2999999999, 12999999999, 26889999999, 37888999999, 5899999999, 3999999999, 13999999999, 36889999999, 25899999999, 8888999999, 18888999999, 7889999999, 4999999999, 7799999999, 17799999999, 6899999999, 16899999999, 27799999999, 38888999999, 26899999999, 37889999999, 5999999999, 15999999999, 56889999999, 36899999999, 25999999999, 8889999999, 18889999999, 7899999999, 17899999999, 28889999999, 46899999999, 6999999999, 16999999999, 58888999999, 38889999999, 26999999999, 37899999999, 57799999999, 88888889999, 56899999999, 36999999999, 48889999999, 8899999999, 18899999999, 7999999999, 17999999999, 28899999999, 46999999999, 27999999999, 127999999999, 58889999999, 38899999999, 57899999999, 37999999999, 137999999999, 88888899999, 56999999999, 78888999999, 48899999999, 8999999999, 18999999999, 68889999999, 168889999999, 28999999999, 128999999999, 268889999999, 378888999999, 58899999999, 38999999999, 57999999999, 157999999999, 258899999999, 88888999999, 188888999999, 78889999999, 48999999999, 9999999999, 19999999999, 68899999999, 168899999999, 29999999999, 129999999999, 268899999999, 378889999999, 58999999999, 39999999999, 139999999999, 368899999999, 258999999999, 88889999999, 188889999999, 78899999999, 49999999999, 77999999999, 177999999999, 68999999999, 168999999999, 277999999999, 388889999999, 268999999999, 378899999999, 59999999999,

159999999999, 568899999999, 368999999999, 259999999999, 88899999999, 188899999999, 78999999999, 178999999999, 288899999999, 468999999999, 69999999999, 169999999999, 588889999999, 388899999999, 269999999999, 378999999999, 577999999999, 888888899999, 568999999999, 369999999999, 488899999999, 88999999999, 188999999999, 79999999999, 179999999999, 288999999999, 469999999999, 279999999999, 1279999999999, 588899999999, 388999999999, 578999999999, 379999999999, 1379999999999, 888888999999, 569999999999, 788889999999, 488999999999, 89999999999, 189999999999, 688899999999, 1688899999999, 289999999999, 1289999999999, 2688899999999, 3788889999999, 588999999999, 389999999999, 579999999999, 1579999999999, 2588999999999, 888889999999, 1888889999999, 788899999999, 489999999999, 99999999999, 199999999999, 688999999999, 1688999999999, 299999999999, 1299999999999, 2688999999999, 3788899999999, 589999999999, 399999999999]

We now find the biggest number with a given digit.

```
> fini := false;
  for n from 900 to 100 by -1 while fini = false do
  if A[n] mod 10 < 9 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$448, 8888888 \tag{19}$$

```
> A[449]
```

$$5689999 \tag{20}$$

```
> fini := false;
  for n from 900 to 100 by -1 while fini = false do
  if A[n] mod 10 < 8 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$151, 2777 \tag{21}$$

```
> fini := false;
  for n from 900 to 10 by -1 while fini = false do
  if A[n] mod 10 < 7 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$112, 2666 \tag{22}$$

```
> fini := false;
  for n from 900 to 10 by -1 while fini = false do
  if A[n] mod 10 < 6 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$48, 444 \tag{23}$$

```
> fini := false;
  for n from 900 to 10 by -1 while fini = false do
  if A[n] mod 10 < 5 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$48, 444 \tag{24}$$

```
> fini := false;
  for n from 900 to 10 by -1 while fini = false do
  if A[n] mod 10 < 4 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$23, 1233 \tag{25}$$

```
> fini := false;
  for n from 900 to 10 by -1 while fini = false do
  if A[n] mod 10 < 3 then print(n, A[n]); fini := true; end if;
  end do;
```

$$fini := false$$

$$12, 222 \tag{26}$$

In particular, if n>448, then L(n) must have the digit 9 in it.

```
> minimalN := proc(n)
  local q, r, k, ans;
  global lowS;;
   if n < 487 then ans := lowS[n];
  else
  q := iquo(n, 81,'r');
  ans := lowS[n - (q - 5)·81]·10^(q - 5) + (10^(q - 5) - 1);
  end if;
  ans;
  end;
```

```
minimalN := proc(n)
    local q, r, k, ans;
    global lowS;
    if n < 487 then
        ans := lowS[n]
    else
        q := iquo(n, 81, 'r'); ans := lowS[n - 81 * q + 405] * 10^(q - 5) + 10^(q - 5) - 1
    end if;
    ans
end proc
```

(27)

```
> minimalbigN := proc(n)
  local q, r, k, ans;
  global lowS;;
   if n < 487 then ans := lowS[n];
  else
  if n < 1000 then
  q := iquo(n, 81,'r');
  ans := lowS[n - (q - 5)·81]·10^(q - 5) + (10^(q - 5) - 1);
  else
  q := iquo(n, 81,'r');
  ans := [ lowS[n - (q - 5)·81], q - 5];
  end if; end if;
  ans;
  end;
```

(28)

```
minimalbigN := proc(n)
    local q, r, k, ans;
    global lowS;
    if n < 487 then
        ans := lowS[n]
    else
        if n < 1000 then
            q := iquo(n, 81, 'r'); ans := lowS[n - 81*q + 405]*10^(q - 5) + 10
            ^(q - 5) - 1
        else
            q := iquo(n, 81, 'r'); ans := [lowS[n - 81*q + 405], q - 5]
        end if
    end if;
    ans
end proc
```

(28)

```
> lowS := A;
```

lowS := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24, 124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6, 16, 116, 1116, 26, 45, 145, 335, 226, 36, 136, 1136, 444, 7, 17, 117, 46, 27, 127, 1127, 246, 227, 37, 137, 1137, 56, 156, 1156, 8, 18, 118, 337, 28, 128, 356, 1356, 66, 38, 57, 157, 266, 238, 257, 1257, 48, 9, 19, 119, 248, 29, 129, 1129, 466, 58, 39, 139, 1139, 258, 239, 1239, 448, 49, 77, 177, 68, 168, 277, 1277, 268, 458, 59, 159, 666, 368, 259, 1259, 2666, 78, 178, 359, 468, 69, 169, 1169, 2468, 269, 378, 577, 1577, 568, 369, 1369, 88, 188, 79, 179, 288, 469, 279, 1279, 668, 388, 578, 379, 1379, 2388, 569, 1569, 488, 89, 189, 777, 1777, 289, 1289, 2777, 4668, 588, 389, 579, 1579, 2588, 2389, 2579, 4488, 489, 99, 199, 688, 1688, 299, 1299, 2688, 4588, 589, 399, 1399, 3688, 2589, 2399, 12399, 788, 499, 779, 1779, 689, 1689, 2779, 12779, 2689, 3788, 599, 1599, 5688, 3689, 2599, 888, 1888, 789, 1789, 2888, 4689, 699, 1699, 6688, 3888, 2699, 3789, 5779, 15779, 5689, 3699, 4888, 889, 1889, 799, 1799, 2889, 4699, 2799, 12799, 5888, 3889, 5789, 3799, 13799, 23889, 5699, 15699, 4889, 899, 1899, 6888, 16888, 2899, 12899, 26888, 45888, 5889, 3899, 5799, 15799, 25889, 23899, 25799, 7888, 4899, 999, 1999, 6889, 16889, 2999, 12999, 26889, 37888, 5899, 3999, 13999, 36889, 25899, 8888, 18888, 7889, 4999, 7799, 17799, 6899, 16899, 27799, 38888, 26899, 37889, 5999, 15999, 56889, 36899, 25999, 8889, 18889, 7899, 17899, 28889, 46899, 6999, 16999, 58888, 38889, 26999, 37899, 57799, 157799, 56899, 36999, 48889, 8899, 18899, 7999, 17999, 28899, 46999, 27999, 127999, 58889, 38899, 57899, 37999, 137999, 238899, 56999, 78888, 48899, 8999, 18999, 68889, 168889, 28999, 128999, 268889, 378888, 58899, 38999, 57999, 157999, 258899, 88888, 188888, 78889, 48999, 9999, 19999, 68899, 168899, 29999, 129999, 268899, 378889, 58999, 39999, 139999, 368899, 258999, 88889, 188889, 78899, 49999, 77999, 177999, 68999, 168999, 277999, 388889, 268999, 378899, 59999, 159999, 568899, 368999, 259999, 88899, 188899, 78999, 178999, 288899, 468999, 69999, 169999, 588889, 388899, 269999, 378999, 577999, 1577999, 568999, 369999, 488899, 88999, 188999, 79999, 179999,

(29)

288999, 469999, 279999, 1279999, 588899, 388999, 578999, 379999, 1379999, 888888, 569999, 788889, 488999, 89999, 189999, 688899, 1688899, 289999, 1289999, 2688899, 3788889, 588999, 389999, 579999, 1579999, 2588999, 888889, 1888889, 788899, 489999, 99999, 199999, 688999, 1688999, 299999, 1299999, 2688999, 3788899, 589999, 399999, 1399999, 3688999, 2589999, 888899, 1888899, 788999, 499999, 779999, 1779999, 689999, 1689999, 2779999, 3888899, 2689999, 3788999, 599999, 1599999, 5688999, 3689999, 2599999, 888999, 1888999, 789999, 1789999, 2888999, 4689999, 699999, 1699999, 5888899, 3888999, 2699999, 3789999, 5779999, 8888888, 5689999, 3699999, 4888999, 889999, 1889999, 799999, 1799999, 2889999, 4699999, 2799999, 12799999, 5888999, 3889999, 5789999, 3799999, 13799999, 8888889, 5699999, 7888899, 4889999, 899999, 1899999, 6888999, 16888999, 2899999, 12899999, 26888999, 37888899, 5889999, 3899999, 5799999, 15799999, 25889999, 8888899, 18888899, 7888999, 4899999, 999999, 1999999, 6889999, 16889999, 2999999, 12999999, 26889999, 37888999, 5899999, 3999999, 13999999, 36889999, 25899999, 8888999, 18888999, 7889999, 4999999, 7799999, 17799999, 6899999, 16899999, 27799999, 38888999, 26899999, 37889999, 5999999, 15999999, 56889999, 36899999, 25999999, 8889999, 18889999, 7899999, 17899999, 28889999, 46899999, 6999999, 16999999, 58888999, 38889999, 26999999, 37899999, 57799999, 88888889, 56899999, 36999999, 48889999, 8899999, 18899999, 7999999, 17999999, 28899999, 46999999, 27999999, 127999999, 58889999, 38899999, 57899999, 37999999, 137999999, 88888899, 56999999, 78888999, 48899999, 8999999, 18999999, 68889999, 168889999, 28999999, 128999999, 268889999, 378888999, 58899999, 38999999, 57999999, 157999999, 258899999, 88888999, 188888999, 78889999, 48999999, 9999999, 19999999, 68899999, 168899999, 29999999, 129999999, 268899999, 378889999, 58999999, 39999999, 139999999, 368899999, 258999999, 88889999, 188889999, 78899999, 49999999, 77999999, 177999999, 68999999, 168999999, 277999999, 388889999, 268999999, 378899999, 59999999, 159999999, 568899999, 368999999, 259999999, 88899999, 188899999, 78999999, 178999999, 288899999, 468999999, 69999999, 169999999, 588889999, 388899999, 269999999, 378999999, 577999999, 888888899, 568999999, 369999999, 488899999, 88999999, 188999999, 79999999, 179999999, 288999999, 469999999, 279999999, 1279999999, 588899999, 388999999, 578999999, 379999999, 1379999999, 888888999, 569999999, 788889999, 488999999, 89999999, 189999999, 688899999, 1688899999, 289999999, 1289999999, 2688899999, 3788889999, 588999999, 389999999, 579999999, 1579999999, 2588999999, 888889999, 1888889999, 788899999, 489999999, 99999999, 199999999, 688999999, 1688999999, 299999999, 1299999999, 2688999999, 3788899999, 589999999, 399999999, 1399999999, 3688999999, 2589999999, 888899999, 1888899999, 788999999, 499999999, 779999999, 1779999999, 689999999, 1689999999, 2779999999, 3888899999, 2689999999, 3788999999, 599999999, 1599999999, 5688999999, 3689999999, 2599999999, 888999999, 1888999999, 789999999, 1789999999, 2888999999, 4689999999, 699999999, 1699999999, 5888899999, 3888999999, 2699999999, 3789999999, 5779999999, 8888888999, 5689999999, 3699999999, 4888999999, 889999999, 1889999999, 799999999, 1799999999, 2889999999,

4699999999, 2799999999, 12799999999, 5888999999, 3889999999, 5789999999, 3799999999, 13799999999, 8888889999, 5699999999, 7888899999, 4889999999, 899999999, 1899999999, 6888999999, 16888999999, 2899999999, 12899999999, 26888999999, 37888899999, 5889999999, 3899999999, 5799999999, 15799999999, 25889999999, 8888899999, 18888899999, 7888999999, 4899999999, 999999999, 1999999999, 6889999999, 16889999999, 2999999999, 12999999999, 26889999999, 37888999999, 5899999999, 3999999999, 13999999999, 36889999999, 25899999999, 8888999999, 18888999999, 7889999999, 4999999999, 7799999999, 17799999999, 6899999999, 16899999999, 27799999999, 38888999999, 26899999999, 37889999999, 5999999999, 15999999999, 56889999999, 36899999999, 25999999999, 8889999999, 18889999999, 7899999999, 17899999999, 28889999999, 46899999999, 6999999999, 16999999999, 58888999999, 38889999999, 26999999999, 37899999999, 57799999999, 88888889999, 56899999999, 36999999999, 48889999999, 8899999999, 18899999999, 7999999999, 17999999999, 28899999999, 46999999999, 27999999999, 127999999999, 58889999999, 38899999999, 57899999999, 37999999999, 137999999999, 88888899999, 56999999999, 78888999999, 48899999999, 8999999999, 18999999999, 68889999999, 168889999999, 28999999999, 128999999999, 268889999999, 378888999999, 58899999999, 38999999999, 57999999999, 157999999999, 258899999999, 88888999999, 188888999999, 78889999999, 48999999999, 9999999999, 19999999999, 68899999999, 168899999999, 29999999999, 129999999999, 268899999999, 378889999999, 58999999999, 39999999999, 139999999999, 368899999999, 258999999999, 88889999999, 188889999999, 78899999999, 49999999999, 77999999999, 177999999999, 68999999999, 168999999999, 277999999999, 388889999999, 268999999999, 378899999999, 59999999999, 159999999999, 568899999999, 368999999999, 259999999999, 88899999999, 188899999999, 78999999999, 178999999999, 288899999999, 468999999999, 69999999999, 169999999999, 588889999999, 388899999999, 269999999999, 378999999999, 577999999999, 888888899999, 568999999999, 369999999999, 488899999999, 88999999999, 188999999999, 79999999999, 179999999999, 288999999999, 469999999999, 279999999999, 1279999999999, 588899999999, 388999999999, 578999999999, 379999999999, 1379999999999, 888888999999, 569999999999, 788889999999, 488999999999, 89999999999, 189999999999, 688899999999, 1688899999999, 289999999999, 1289999999999, 2688899999999, 3788889999999, 588999999999, 389999999999, 579999999999, 1579999999999, 2588999999999, 888889999999, 1888889999999, 788899999999, 489999999999, 99999999999, 199999999999, 688999999999, 1688999999999, 299999999999, 1299999999999, 2688999999999, 3788899999999, 589999999999, 399999999999]

We now verify a result we need for analyzing large values, namely, that for n1>367 and n2>n1+108, then L(n2) >= L(n1) + 431000.

```
> for b1 from 350 to 2000 do
  for b2 from b1 + 108 to 2000 do
```

```
if minimalN(b2) − minimalN(b1) < 1.5·10^6 then print(b1, b2, b2 − b1, minimalN(b2)
    − minimalN(b1)) end if;
end do;
end do:
```

350, 469, 119, 740000
350, 486, 136, 840000
351, 469, 118, 331100
351, 470, 119, 1331100
351, 486, 135, 431100
351, 487, 136, 1431100
352, 469, 117, 531000
352, 486, 134, 631000
353, 469, 116, 640000
353, 486, 133, 740000
354, 469, 115, 811100
354, 486, 132, 911100
355, 469, 114, 711100
355, 486, 131, 811100
356, 469, 113, 821000
356, 486, 130, 921000
357, 469, 112, 721000
357, 486, 129, 821000
358, 469, 111, 611100
358, 486, 128, 711100
359, 469, 110, 431000
359, 470, 111, 1431000
359, 486, 127, 531000
360, 469, 109, 830000
360, 486, 126, 930000
361, 469, 108, 730000
361, 486, 125, 830000
362, 470, 108, 1311110
362, 486, 124, 411110
362, 487, 125, 1411110
363, 486, 123, 611100
364, 486, 122, 730000
365, 486, 121, 621000
366, 486, 120, 422000
366, 487, 121, 1422000
367, 486, 119, -578000
367, 487, 120, 422000
367, 490, 123, 1422000
368, 486, 118, 431000

368, 487, 119, 1431000
369, 486, 117, 630000
370, 486, 116, 511100
371, 486, 115, 911000
372, 486, 114, 811000
373, 486, 113, 920000
374, 486, 112, 820000
375, 486, 111, 711000
376, 486, 110, 530000
377, 486, 109, 720000
378, 486, 108, -280000
378, 487, 109, 720000
379, 487, 108, 1411100
448, 567, 119, 1111111
459, 567, 108, -2800000
540, 648, 108, -28000000
621, 729, 108, -280000000
702, 810, 108, -2800000000
783, 891, 108, -28000000000
864, 972, 108, -280000000000
945, 1053, 108, -2800000000000
1026, 1134, 108, -28000000000000
1107, 1215, 108, -280000000000000
1188, 1296, 108, -2800000000000000
1269, 1377, 108, -28000000000000000
1350, 1458, 108, -280000000000000000
1431, 1539, 108, -2800000000000000000
1512, 1620, 108, -28000000000000000000
1593, 1701, 108, -280000000000000000000
1674, 1782, 108, -2800000000000000000000
1755, 1863, 108, -28000000000000000000000
1836, 1944, 108, -280000000000000000000000 **(30)**

> *minimalN*(367) ; *minimalN*(486);

1577999
999999 **(31)**

> *minimalN*(368); *minimalN*(486); % − %%

568999
999999
431000 **(32)**

We next analyze another result we need for large values, namely, that for n1 > 20, if L(n1) > L(n2) then L(L(n1)) > L(L(n2))+10^6. we already know that if the values are over 378 and the difference over 108, then we are OK.

> **for** *n1* **from** 200 **to** 700 **do**

```
if minimalN(n1) < 378 then print('n1', n1) else
 for n2 from n1 + 1 to 701 do
  if minimalN(n2) < 378 then print('n2', n2) else
 if abs(minimalN(n1) - minimalN(n2) ) < 120 then print(n1, n2, minimalN(n1)
      - minimalN(n2), minimalN(n1) , minimalN(n2), minimalbigN(minimalN(n1)),
      minimalbigN(minimalN(n2))); end if;
 end if;
 end do;
 end if;
 end do:
```

201, 203, 99, 3888, 3789, [99999, 43], [4889999, 41]
201, 218, -1, 3888, 3889, [99999, 43], [199999, 43]
201, 220, 89, 3888, 3799, [99999, 43], [3899999, 41]
201, 235, -11, 3888, 3899, [99999, 43], [3688999, 43]
201, 252, -111, 3888, 3999, [99999, 43], [888999, 44]
202, 215, -100, 2699, 2799, [1599999, 28], [3699999, 29]
203, 207, 90, 3789, 3699, [4889999, 41], [12799999, 40]
203, 218, -100, 3789, 3889, [4889999, 41], [199999, 43]
203, 220, -10, 3789, 3799, [4889999, 41], [3899999, 41]
203, 235, -110, 3789, 3899, [4889999, 41], [3688999, 43]
204, 206, 90, 5779, 5689, [3689999, 66], [689999, 65]
204, 217, -109, 5779, 5888, [3689999, 66], [3889999, 67]
204, 219, -10, 5779, 5789, [3689999, 66], [5888899, 66]
204, 223, 80, 5779, 5699, [3689999, 66], [2599999, 65]
204, 234, -110, 5779, 5889, [3689999, 66], [5789999, 67]
204, 236, -20, 5779, 5799, [3689999, 66], [1889999, 66]
205, 224, 80, 15779, 15699, [1899999, 189], [6888999, 188]
205, 237, -20, 15779, 15799, [1899999, 189], [299999, 190]
206, 219, -100, 5689, 5789, [689999, 65], [5888899, 66]
206, 223, -10, 5689, 5699, [689999, 65], [2599999, 65]
206, 236, -110, 5689, 5799, [689999, 65], [1889999, 66]
207, 220, -100, 3699, 3799, [12799999, 40], [3899999, 41]
208, 225, -1, 4888, 4889, [3689999, 55], [2599999, 55]
208, 242, -11, 4888, 4899, [3689999, 55], [3888999, 55]
208, 259, -111, 4888, 4999, [3689999, 55], [3799999, 56]
209, 211, 90, 889, 799, 788899999999, 268889999999
209, 226, -10, 889, 899, 788899999999, 589999999999
209, 243, -110, 889, 999, 788899999999, 56889999999999
210, 212, 90, 1889, 1799, [1599999, 18], [779999, 17]
210, 227, -10, 1889, 1899, [1599999, 18], [699999, 18]
210, 244, -110, 1889, 1999, [1599999, 18], [5888999, 19]
211, 226, -100, 799, 899, 268889999999, 589999999999
212, 227, -100, 1799, 1899, [779999, 17], [699999, 18]
213, 215, 90, 2889, 2799, [12799999, 30], [3699999, 29]

213, 230, −10, 2889, 2899, [12799999, 30], [899999, 30]
213, 247, −110, 2889, 2999, [12799999, 30], [688999, 32]
215, 230, −100, 2799, 2899, [3699999, 29], [899999, 30]
216, 231, −100, 12799, 12899, [199999, 153], [1689999, 154]
217, 219, 99, 5888, 5789, [3889999, 67], [5888899, 66]
217, 234, −1, 5888, 5889, [3889999, 67], [5789999, 67]
217, 236, 89, 5888, 5799, [3889999, 67], [1889999, 66]
217, 251, −11, 5888, 5899, [3889999, 67], [16888999, 67]
217, 268, −111, 5888, 5999, [3889999, 67], [1299999, 69]
218, 220, 90, 3889, 3799, [199999, 43], [3899999, 41]
218, 235, −10, 3889, 3899, [199999, 43], [3688999, 43]
218, 252, −110, 3889, 3999, [199999, 43], [888999, 44]
219, 223, 90, 5789, 5699, [5888899, 66], [2599999, 65]
219, 234, −100, 5789, 5889, [5888899, 66], [5789999, 67]
219, 236, −10, 5789, 5799, [5888899, 66], [1889999, 66]
219, 251, −110, 5789, 5899, [5888899, 66], [16888999, 67]
220, 235, −100, 3799, 3899, [3899999, 41], [3688999, 43]
222, 239, −10, 23889, 23899, [15799999, 289], [299999, 290]
223, 236, −100, 5699, 5799, [2599999, 65], [1889999, 66]
224, 237, −100, 15699, 15799, [6888999, 188], [299999, 190]
225, 242, −10, 4889, 4899, [2599999, 55], [3888999, 55]
225, 259, −110, 4889, 4999, [2599999, 55], [3799999, 56]
226, 243, −100, 899, 999, 589999999999, 56889999999999
227, 244, −100, 1899, 1999, [699999, 18], [5888999, 19]
228, 245, −1, 6888, 6889, [1688999, 80], [299999, 80]
228, 262, −11, 6888, 6899, [1688999, 80], [1888899, 80]
228, 279, −111, 6888, 6999, [1688999, 80], [1789999, 81]
229, 246, −1, 16888, 16889, [2699999, 203], [3789999, 203]
229, 263, −11, 16888, 16899, [2699999, 203], [2889999, 203]
229, 280, −111, 16888, 16999, [2699999, 203], [26888999, 204]
230, 247, −100, 2899, 2999, [899999, 30], [688999, 32]
231, 248, −100, 12899, 12999, [1689999, 154], [3888999, 155]
232, 249, −1, 26888, 26889, [8888899, 326], [18888899, 326]
232, 266, −11, 26888, 26899, [8888899, 326], [3788899, 327]
232, 283, −111, 26888, 26999, [8888899, 326], [1599999, 328]
234, 236, 90, 5889, 5799, [5789999, 67], [1889999, 66]
234, 251, −10, 5889, 5899, [5789999, 67], [16888999, 67]
234, 268, −110, 5889, 5999, [5789999, 67], [1299999, 69]
235, 252, −100, 3899, 3999, [3688999, 43], [888999, 44]
236, 251, −100, 5799, 5899, [1889999, 66], [16888999, 67]
238, 240, 90, 25889, 25799, [1799999, 314], [3789999, 313]
238, 255, −10, 25889, 25899, [1799999, 314], [8888889, 314]

238, 272, −110, 25889, 25999, [1799999, 314], [7888999, 315]
240, 255, −100, 25799, 25899, [3789999, 313], [8888889, 314]
241, 258, −1, 7888, 7889, [1888999, 92], [789999, 92]
241, 260, 89, 7888, 7799, [1888999, 92], [2689999, 91]
241, 275, −11, 7888, 7899, [1888999, 92], [5779999, 92]
241, 292, −111, 7888, 7999, [1888999, 92], [5699999, 93]
242, 259, −100, 4899, 4999, [3888999, 55], [3799999, 56]
245, 262, −10, 6889, 6899, [299999, 80], [1888899, 80]
245, 279, −110, 6889, 6999, [299999, 80], [1789999, 81]
246, 263, −10, 16889, 16899, [3789999, 203], [2889999, 203]
246, 280, −110, 16889, 16999, [3789999, 203], [26888999, 204]
249, 266, −10, 26889, 26899, [18888899, 326], [3788899, 327]
249, 283, −110, 26889, 26999, [18888899, 326], [1599999, 328]
250, 267, −1, 37888, 37889, [5699999, 462], [7888899, 462]
250, 284, −11, 37888, 37899, [5699999, 462], [5889999, 462]
250, 301, −111, 37888, 37999, [5699999, 462], [1399999, 464]
251, 268, −100, 5899, 5999, [16888999, 67], [1299999, 69]
254, 271, −10, 36889, 36899, [2888999, 450], [5689999, 450]
254, 288, −110, 36889, 36999, [2888999, 450], [4889999, 451]
255, 272, −100, 25899, 25999, [8888889, 314], [7888999, 315]
256, 273, −1, 8888, 8889, [13799999, 104], [8888889, 104]
256, 290, −11, 8888, 8899, [13799999, 104], [26888999, 104]
256, 307, −111, 8888, 8999, [13799999, 104], [589999, 106]
257, 274, −1, 18888, 18889, [788999, 228], [499999, 228]
257, 291, −11, 18888, 18899, [788999, 228], [1599999, 228]
257, 308, −111, 18888, 18999, [788999, 228], [3699999, 229]
258, 260, 90, 7889, 7799, [789999, 92], [2689999, 91]
258, 275, −10, 7889, 7899, [789999, 92], [5779999, 92]
258, 292, −110, 7889, 7999, [789999, 92], [5699999, 93]
260, 275, −100, 7799, 7899, [2689999, 91], [5779999, 92]
261, 276, −100, 17799, 17899, [8888889, 214], [7888999, 215]
262, 279, −100, 6899, 6999, [1888899, 80], [1789999, 81]
263, 280, −100, 16899, 16999, [2889999, 203], [26888999, 204]
265, 282, −1, 38888, 38889, [589999, 475], [399999, 475]
265, 299, −11, 38888, 38899, [589999, 475], [689999, 475]
265, 316, −111, 38888, 38999, [589999, 475], [5888899, 476]
266, 283, −100, 26899, 26999, [3788899, 327], [1599999, 328]
267, 284, −10, 37889, 37899, [7888899, 462], [5889999, 462]
267, 301, −110, 37889, 37999, [7888899, 462], [1399999, 464]
270, 287, −10, 56889, 56899, [5688999, 697], [1699999, 697]
270, 304, −110, 56889, 56999, [5688999, 697], [3889999, 698]
271, 288, −100, 36899, 36999, [5689999, 450], [4889999, 451]

273, 290, −10, 8889, 8899, [8888889, 104], [26888999, 104]
273, 307, −110, 8889, 8999, [8888889, 104], [589999, 106]
274, 291, −10, 18889, 18899, [499999, 228], [1599999, 228]
274, 308, −110, 18889, 18999, [499999, 228], [3699999, 229]
275, 292, −100, 7899, 7999, [5779999, 92], [5699999, 93]
276, 293, −100, 17899, 17999, [7888999, 215], [779999, 217]
277, 294, −10, 28889, 28899, [2799999, 351], [4889999, 351]
277, 311, −110, 28889, 28999, [2799999, 351], [199999, 353]
278, 295, −100, 46899, 46999, [99999, 574], [689999, 575]
281, 298, −1, 58888, 58889, [199999, 722], [688999, 722]
281, 315, −11, 58888, 58899, [199999, 722], [2589999, 722]
281, 332, −111, 58888, 58999, [199999, 722], [1888999, 723]
282, 299, −10, 38889, 38899, [399999, 475], [689999, 475]
282, 316, −110, 38889, 38999, [399999, 475], [5888899, 476]
284, 301, −100, 37899, 37999, [5889999, 462], [1399999, 464]
285, 300, −100, 57799, 57899, [4888999, 708], [1899999, 709]
287, 304, −100, 56899, 56999, [1699999, 697], [3889999, 698]
289, 306, −10, 48889, 48899, [4888999, 598], [3889999, 598]
289, 323, −110, 48889, 48999, [4888999, 598], [15799999, 599]
290, 307, −100, 8899, 8999, [26888999, 104], [589999, 106]
291, 308, −100, 18899, 18999, [1599999, 228], [3699999, 229]
294, 311, −100, 28899, 28999, [4889999, 351], [199999, 353]
298, 315, −10, 58889, 58899, [688999, 722], [2589999, 722]
298, 332, −110, 58889, 58999, [688999, 722], [1888999, 723]
299, 316, −100, 38899, 38999, [689999, 475], [5888899, 476]
300, 317, −100, 57899, 57999, [1899999, 709], [1688999, 711]
305, 322, −1, 78888, 78889, [15799999, 968], [25889999, 968]
305, 339, −11, 78888, 78899, [15799999, 968], [1299999, 969]
305, 356, −111, 78888, 78999, [15799999, 968], [3788999, 970]
306, 323, −100, 48899, 48999, [3889999, 598], [15799999, 599]
309, 326, −10, 68889, 68899, [3888999, 845], [799999, 845]
309, 343, −110, 68889, 68999, [3888999, 845], [2899999, 846]
310, 327, −10, 168889, 168899, [299999, 2080], [1888899, 2080]
310, 344, −110, 168889, 168999, [299999, 2080], [1789999, 2081]
313, 330, −10, 268889, 268899, [1799999, 3314], [8888889, 3314]
313, 347, −110, 268889, 268999, [1799999, 3314], [7888999, 3315]
314, 331, −1, 378888, 378889, [2889999, 4672], [4699999, 4672]
314, 348, −11, 378888, 378899, [2889999, 4672], [7888899, 4672]
314, 365, −111, 378888, 378999, [2889999, 4672], [99999, 4674]
315, 332, −100, 58899, 58999, [2589999, 722], [1888999, 723]
319, 336, −100, 258899, 258999, [2689999, 3191], [5779999, 3192]
320, 337, −1, 88888, 88889, [1888999, 1092], [789999, 1092]

320, 354, −11, 88888, 88899, [1888999, 1092], [5779999, 1092]
320, 371, −111, 88888, 88999, [1888999, 1092], [5699999, 1093]
321, 338, −1, 188888, 188889, [8888899, 2326], [18888899, 2326]
321, 355, −11, 188888, 188899, [8888899, 2326], [3788899, 2327]
321, 372, −111, 188888, 188999, [8888899, 2326], [1599999, 2328]
322, 339, −10, 78889, 78899, [25889999, 968], [1299999, 969]
322, 356, −110, 78889, 78999, [25889999, 968], [3788999, 970]
326, 343, −100, 68899, 68999, [799999, 845], [2899999, 846]
327, 344, −100, 168899, 168999, [1888899, 2080], [1789999, 2081]
330, 347, −100, 268899, 268999, [8888889, 3314], [7888999, 3315]
331, 348, −10, 378889, 378899, [4699999, 4672], [7888899, 4672]
331, 365, −110, 378889, 378999, [4699999, 4672], [99999, 4674]
335, 352, −100, 368899, 368999, [599999, 4549], [5689999, 4550]
337, 354, −10, 88889, 88899, [789999, 1092], [5779999, 1092]
337, 371, −110, 88889, 88999, [789999, 1092], [5699999, 1093]
338, 355, −10, 188889, 188899, [18888899, 2326], [3788899, 2327]
338, 372, −110, 188889, 188999, [18888899, 2326], [1599999, 2328]
339, 356, −100, 78899, 78999, [1299999, 969], [3788999, 970]
346, 363, −10, 388889, 388899, [589999, 4796], [1779999, 4796]
346, 380, −110, 388889, 388999, [589999, 4796], [1699999, 4797]
348, 365, −100, 378899, 378999, [7888899, 4672], [99999, 4674]
351, 368, −100, 568899, 568999, [699999, 7018], [5888999, 7019]
354, 371, −100, 88899, 88999, [5779999, 1092], [5699999, 1093]
355, 372, −100, 188899, 188999, [3788899, 2327], [1599999, 2328]
358, 375, −100, 288899, 288999, [2799999, 3561], [5889999, 3562]
362, 379, −10, 588889, 588899, [689999, 7265], [2599999, 7265]
362, 396, −110, 588889, 588999, [689999, 7265], [1889999, 7266]
363, 380, −100, 388899, 388999, [1779999, 4796], [1699999, 4797]
370, 387, −100, 488899, 488999, [899999, 6030], [688999, 6032]
379, 396, −100, 588899, 588999, [2599999, 7265], [1889999, 7266]
384, 401, −1, 888888, 888889, [15799999, 10968], [25889999, 10968]
384, 418, −11, 888888, 888899, [15799999, 10968], [1299999, 10969]
384, 435, −111, 888888, 888999, [15799999, 10968], [3788999, 10970]
386, 403, −10, 788889, 788899, [888999, 9734], [2699999, 9734]
386, 420, −110, 788889, 788999, [888999, 9734], [13799999, 9735]
390, 407, −100, 688899, 688999, [15799999, 8499], [888899, 8501]
391, 408, −100, 1688899, 1688999, [799999, 20845], [2899999, 20846]
394, 411, −100, 2688899, 2688999, [2689999, 33191], [5779999, 33192]
395, 412, −10, 3788889, 3788899, [1789999, 46771], [8888888, 46771]
395, 429, −110, 3788889, 3788999, [1789999, 46771], [7888899, 46772]
401, 418, −10, 888889, 888899, [25889999, 10968], [1299999, 10969]
401, 435, −110, 888889, 888999, [25889999, 10968], [3788999, 10970]

402, 419, −10, 1888889, 1888899, [1799999, 23314], [8888889, 23314]
402, 436, −110, 1888889, 1888999, [1799999, 23314], [7888999, 23315]
403, 420, −100, 788899, 788999, [2699999, 9734], [13799999, 9735]
412, 429, −100, 3788899, 3788999, [8888888, 46771], [7888899, 46772]
418, 435, −100, 888899, 888999, [1299999, 10969], [3788999, 10970]
419, 436, −100, 1888899, 1888999, [8888889, 23314], [7888999, 23315]
427, 444, −100, 3888899, 3888999, [589999, 48006], [5688999, 48007]
443, 460, −100, 5888899, 5888999, [1699999, 72697], [3889999, 72698]
448, 465, −1, 8888888, 8888889, [2599999, 109734], [888999, 109734]
448, 482, −11, 8888888, 8888899, [2599999, 109734], [2699999, 109734]
448, 499, −111, 8888888, 8888999, [2599999, 109734], [13799999, 109735]
465, 482, −10, 8888889, 8888899, [888999, 109734], [2699999, 109734]
465, 499, −110, 8888889, 8888999, [888999, 109734], [13799999, 109735]
467, 484, −100, 7888899, 7888999, [6888999, 97388], [299999, 97390]
476, 493, −100, 37888899, 37888999, [788999, 467759], [2888999, 467760]
482, 499, −100, 8888899, 8888999, [2699999, 109734], [13799999, 109735]
483, 500, −100, 18888899, 18888999, [2689999, 233191], [5779999, 233192]
529, 546, −10, 88888889, 88888899, [3889999, 1097388], [6888999, 1097388]
529, 563, −110, 88888889, 88888999, [3889999, 1097388], [299999, 1097390]
546, 563, −100, 88888899, 88888999, [6888999, 1097388], [299999, 1097390]
610, 627, −100, 888888899, 888888999, [688999, 10973932], [2779999, 10973933] **(33)**

> $minimalN(448); minimalN(465);$

8888888
8888889 **(34)**

> $minimalbigN(minimalN(448)); minimalbigN(minimalN(465)); op(2, \%);$

[2599999, 109734]
[888999, 109734]
109734 **(35)**

Unfortunately, the minimalbigN module was badly defined with an inconsistent datatype. So we redefine it here for the next calculation.

```
> NEWminimalbigN := proc(n)
  local q, r, k, ans;
  global lowS;;
   if n < 487 then ans := [minimalN(n), 0];
  else
  q := iquo(n, 81,'r');
  ans := [ lowS[n − (q − 5)·81], q − 5];
  end if;
  ans;
  end;
```

```
NEWminimalbigN := proc(n)
    local q, r, k, ans;
    global lowS;
    if n < 487 then
```

```
        ans := [minimalN(n), 0]
    else
        q := iquo(n, 81, 'r'); ans := [lowS[n - 81*q + 405], q - 5]
    end if;
    ans
end proc
```

```
> NEWminimalbigN(265); NEWminimalbigN(678);
```

$$[38888, 0]$$

$$[888999, 3] \tag{37}$$

```
> T := 1000;
 for n1 from 300 to T do
 for n2 from n1 + 1 to T + 1 do
 a := NEWminimalbigN(n1);
 b := NEWminimalbigN(n2);
 if  op(2, a ) = op( 2, b) then
 if op(1, a) > op(1, b) and minimalN(op(1, a))-10^6 < minimalN(op(1, b)) then
print(n1, n2, a, b, op(1, a)-op(1, b)) end if;
if op(1, b) > op(1, a) and minimalN(op(1, b))-10^6 < minimalN(op(1, a)) then
print(n1, n2, a, b, op(1, a)-op(1, b)) end if;
end if;
 end do: end do:
```

$$T := 1000$$

$$305, 339, [78888, 0], [78899, 0], -11$$
$$309, 326, [68889, 0], [68899, 0], -10$$
$$320, 337, [88888, 0], [88889, 0], -1$$
$$322, 339, [78889, 0], [78899, 0], -10$$
$$384, 418, [888888, 0], [888899, 0], -11$$
$$401, 418, [888889, 0], [888899, 0], -10$$
$$448, 465, [8888888, 0], [8888889, 0], -1$$
$$529, 546, [8888888, 1], [8888889, 1], -1$$
$$610, 627, [8888888, 2], [8888889, 2], -1$$
$$691, 708, [8888888, 3], [8888889, 3], -1$$
$$772, 789, [8888888, 4], [8888889, 4], -1$$
$$853, 870, [8888888, 5], [8888889, 5], -1$$
$$934, 951, [8888888, 6], [8888889, 6], -1 \tag{38}$$

```
> T := 1000;
 for n1 from 300 to T do
 for n2 from n1 + 1 to T + 1 do
 a := NEWminimalbigN(n1);
 b := NEWminimalbigN(n2);
 if evalf(log10(op(1, a)) + op(2, a)) > evalf(log10(op(1, b)) + op(2, b)) and  then
print(n1, n2, a, b, op(1, a)-op(1, b)) end if;
if op(1, b) > op(1, a) and minimalN(op(1, b))-10^6 < minimalN(op(1, a)) then
print(n1, n2, a, b, op(1, a)-op(1, b)) end if;
  end do: end do:
```

$$100$$

$$108 \tag{39}$$

> $NEWminimalbigN(minimalN(106)); minimalN(106);$

$$[137, 0]$$

$$59 \tag{40}$$

> $NEWminimalbigN(68);$

$$[28, 0] \tag{41}$$

> $onestep(28)$

$$68 \tag{42}$$

>